WO0179994

Publication Title:

SYSTEM AND METHOD FOR DYNAMICALLY MANAGING ELECTRONIC BUSINESS PROCESS

Abstract:

Abstract of WO0179994

A business process engine hosts the execution of electronic business processes and generates event objects for communicating the state of business processes to a business rule engine. The business rule engine hosts the execution of business rule functions triggered dynamically in response to the event objects. Business rule functions exercise control over the business process engine or external systems, generate business metrics, react to business metrics and adjust parameters of business metrics and other rule functions. Business rule functions and business metrics may also generate event objects. A business management engine hosts the generation of business reports reflecting system status. Additionally, the business management engine receives user input manually creating, defining or re-defining business rule functions, metrics and business report format. Business rule functions, metrics and business report are created or defined during the real-time operation of the EBMS without interrupting electronic business process flow within the business process engine.

Data supplied from the esp@cenet database - Worldwide

Courtesy of http://v3.espacenet.com

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 October 2001 (25.10.2001)

PCT

(10) International Publication Number
WO 01/79994 A2

(51) International Patent Classification⁷: G06F 9/00

(21) International Application Number: PCT/US01/40502

(22) International Filing Date: 12 April 2001 (12.04.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/197,293 14 April 2000 (14.04.2000) US

(71) Applicant *(for all designated States except US)*: **SAVVION INCORPORATED** [US/US]; 5000 Old Ironsides Drive, Santa Clara, CA 95054 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **DURAND, Jacques, R.** [US/US]; 1685 Alby Court, San Jose, CA 95124 (US). **DESAI, Shvetal, T.** [IN/US]; 3480 Granada Avenue, #294, Santa Clara, CA 95051 (US). **MAKHFI, Pejman** [IR/US]; 5808 Comanche Drive, San Jose, CA 95123 (US). **GARIMELLA, Phanendra, B.** [IN/US]; 471 Acalanes, #45, Sunnyvale, CA 94086 (US). **LIU, Kelvin, W.** [US/US]; 20373 Via Napoli, Cupertino, CA 95014 (US).

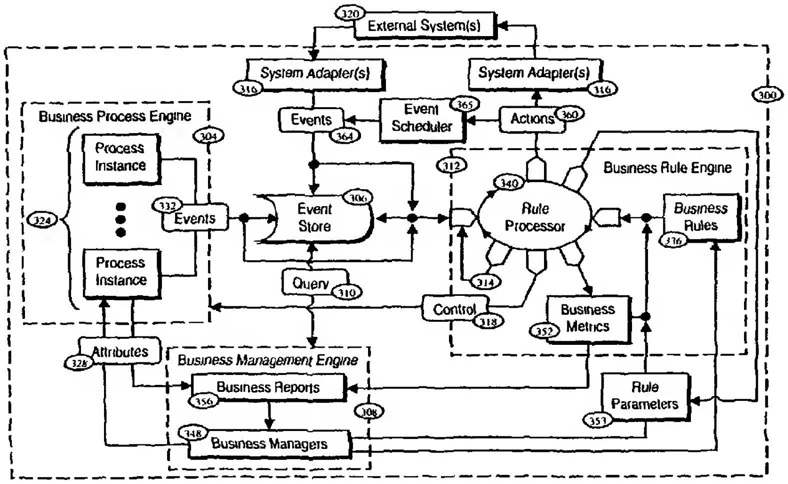
(74) Agents: **SYROWIK, David, R.** et al.; Brooks & Kushman P.C., 22nd Floor, 1000 Town Center, Southfield, MI 48075 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR DYNAMICALLY MANAGING ELECTRONIC BUSINESS PROCESS



(57) **Abstract:** A business process engine hosts the execution of electronic business processes and generates event objects for communicating the state of business processes to a business rule engine. The business rule engine hosts the execution of business rule functions triggered dynamically in response to the event objects. Business rule functions exercise control over the business process engine or external systems, generate business metrics, react to business metrics and adjust parameters of business metrics and other rule functions. Business rule functions and business metrics may also generate event objects. A business management engine hosts the generation of business reports reflecting system status. Additionally, the business management engine receives user input manually creating, defining or re-defining business rule functions, metrics and business report format. Business rule functions, metrics and business report are created or defined during the real-time operation of the EBMS without interrupting electronic business process flow within the business process engine.

Published:

— *without international search report and to be republished upon receipt of that report*

SYSTEM AND METHOD FOR DYNAMICALLY MANAGING ELECTRONIC BUSINESS PROCESS

TECHNICAL FIELD

The present invention relates to process management systems generally, and more particularly, to a system and method for dynamically managing electronic business processes.

BACKGROUND ART

Effective management of electronic business processes requires a comprehensive on-line platform capable of real-time dynamic control and reporting of system status. Prior art systems and methodologies, however, lack the ability to provide automated self-management based upon a real-time coordination of business rules, business metrics, and system resources.

For example, prior art electronic business process management systems tend to implement business process flow control based on a rigid array of static rules. Although some prior art systems provide a user-interface for manually defining or editing rule parameters, no system or methodology exists for managing electronic business processes with rules that are dynamically triggered and adapted in an automated fashion based on the critical value of business-critical metrics such as business transaction volume, response time, turn-around time, event priority, etc.

Conventionally, existing systems tend to generate system status reports and await user analysis and response before reacting to changing environmental conditions such as resource usage or client demand. What is needed is a method and system possessing functionality to automatically administer necessary business process (*i.e.*, business process timing and attributes) during real-time system operation based on dynamically-generated event objects and system metrics.

Another disadvantage of prior art electronic business control systems is their inability to adjust business rules "on-the-fly" without interfering with business process flow. Such functionality, however, is critical in an on-line environment where clients continuously request system response and process performance.

Yet another drawback to prior art electronic business management systems is their failure to define business rule parameters and business metrics in a standard format such that they may be interchangeably used by rules for decisions, as well as updated by rules in response to internal or external events. Additionally, business metrics defined by prior art systems lack the ability to independently generate event objects for triggering subsequent system control.

Other disadvantages of prior art systems include their inability to execute business rules in response to events generated externally or in response to a correlation of event objects generated over time. Conventionally, rule-based control systems monitor local system status only at discrete instances of time and lack functionality to sense externally-generated events or correlate event sequences that occur at separate points in time.

What is needed is a method and system for dynamically managing electronic business processes that solves these and other problems associated with prior art electronic business management systems and methodologies.

DISCLOSURE OF INVENTION

One object of the present invention is to provide a system and method for dynamically managing electronic business processes. In carrying out this object and other objects, features and advantages, the present invention includes a business process engine for hosing the execution of a plurality of electronic business process instances. Each process instance, in turn, comprises a plurality of work steps. Before and after the execution of each work step, an event object is generated for communicating state attributes of the process instance or work step to a business rule engine. In an additional or alternate embodiment of the present invention, event objects may be generated by external systems and communicated to the business rule engine via an appropriate external system adapter. Preferably, a persistent storage device buffers event objects generated by the business process engine and external systems for subsequent transmission to the business rule engine.

The business rule engine hosts the execution of a plurality of business rule functions which are triggered dynamically in response to the event objects, separately or in correlation. Upon execution, business rule functions may exercise control over the business process engine or external systems (via appropriate system adapters), generate business metrics, react to business metrics and adjust parameters of business metrics and other rule functions.

Notably, business metrics themselves may be configured to dynamically generate event objects in response to attaining a predefined attribute or parameter value.

A business management engine hosts the generation of business reports reflecting EBMS system status. Additionally, the business management engine receives user input manually creating, defining or re-defining business rule functions, metrics and business report format. Notably, business rule functions, metrics and business report can be created, defined or re-defined during the real-time operation of the EBMS without interrupting electronic business process flow within the business process engine.

In accord with a preferred embodiment of the present invention, the EBMS may be implemented over several computing platforms including stand-alone operating environments and networked environments such as intranets and extranets including the Internet.

The above objects and other objects, features, and advantages of the present invention are readily apparent from the following detailed description of the best mode for carrying out the invention when taken in connection with the accompanying drawings.

BRIEF DESCRIPTION OF DRAWINGS

FIGURE 1 is a block diagram illustrating an overview of the electronic business management capabilities provided by the present invention;

FIGURE 2 is a block flow diagram illustrating an overview of a preferred system for implementing the present invention;

FIGURE 3 is a block flow diagram illustrating a detailed description of a preferred system for implementing the present invention;

FIGURE 4 is a detailed illustration of an electronic business process instance in accord with the present invention;

FIGURE 5 is block diagram illustrating an overview of business rule functions in accord with the present invention;

FIGURE 6 is block diagram illustrating an overview of business rule function definition in accord with the present invention;

FIGURE 7 is block diagram illustrating an overview of event object definition in accord with the present invention;

FIGURE 8 is a block flow diagram illustrating a preferred event correlation sequence within the rule engine where all of the correlated events belong to the same process instance; and

FIGURE 9 is a block flow diagram illustrating a preferred event correlation sequence within the rule engine where all of the events either correlate to a plurality of process instances or originate from external systems.

BEST MODE FOR CARRYING OUT THE INVENTION

Referring to Figure 1, an overview of the EBMS capabilities is shown. The EBMS 100 addresses essential elements of a multi-process electronic business including but not limited to business information 104, resources 108, and operations 112.

Business information 104 includes but is not limited to the state of process instance timing, the state of process instances themselves, the volume of concurrent process instances, process instance profiles, resource usage or availability and external events about the context of the business.

Resources 108 include but are not limited to database activity, third-party software tools, human workforce, computer software, and computer hardware such as computer networks, client computers, and server computers.

Operations 112 include but are not limited to process instance timing, chaining of work steps, resource allocation, prioritization of process instances, external system control, assignment of tasks, monitoring, notification of process instance state, report generation, e-mail, and printing.

Referring to Figure 2, an overview of a preferred embodiment of the EBMS is shown. The EBMS 200 generally comprises a business process engine 204, a business management engine 208, a business rule engine 212, an event store 206, and system adapters 216 for communication with external systems 220.

In accord with a preferred embodiment of the EBMS, the business rule engine 212 is in operable communication with the business management engine 208, the business process engine 204, the event store 206, and system adapters 216. The business process engine is in operable communication with the event store 206, the business management engine 208, and the business rule engine 212. The business management engine 208 is in operable communication with the event store 206, the business process engine 204, and the business rule engine 212. In further accord with the preferred embodiment, the system adapters 216 operably connect the external systems 220, to the event store 206 and the business rule engine 212.

The business process engine 204 may be implemented in intranets, extranets, and the Internet to host and run electronic business processes. In response to the execution of a business process, the business process engine 204 is configured to generate and communicate business process information to the event store 206, the business management engine 208, and the business rule engine 212.

The event store 206 is configured to log, within persistent storage, business process information generated by the business process engine 204.

Generally, the business rule engine 212 pulls business process information stored within the event store 206, the business process engine 204, and system adapters 216. Based on the business process information, the business rule engine 212 implements both operational logic and management polices within the EBMS 200 and external systems 220 via system adapters 216. For implementation, the operational logic and management policies are defined by an encoded rule language residing in the business rule engine 212.

Notably, the business rule engine 212 allows the business management engine 208 to dynamically view, create, or modify operational logic and management policies during the real-time operation of the EBMS 200.

Additionally, the business rule engine 212 is configured to generate and communicate business process information pulled from the event store 206 to the business management engine 208. Essentially, the business process information communicated to the business management engine 208 reflects the comprehensive state of the EBMS 200 in terms of business metrics. In accord with a preferred embodiment of the present invention, the business process information is communicated to the business management engine 208 dynamically during the real-time operation of the EBMS 200.

Generally, the business management engine 208 monitors and analyzes the business process information generated by the event storage 206, business rule engine 212, and the business process engine 204. In response to the analysis (or independently), business managers can create or modify operational logic and management policies for implementation in the business rule engine 208. Additionally, business managers can directly effect control over processes executing within the business process engine 204.

Notably, business managers can view, create, or modify operational logic and management policies, or control an executing process, dynamically during the real-time operation of the EBMS 200 without interruption of business processes executing within the business process engine 204.

The system adapters 216 are configured to operably interface the EBMS 200 with external systems 220 for i) monitoring processes executing external to the EBMS 200, and ii) effecting external activity. Additionally, the external systems 220 may be controlled to perform the tasks of a business process originally internal to the EBMS 200.

EBMS 200 may be implemented on a plurality of computing platforms understood by those of ordinary skill in the art of computing and information system development and management. Computing platforms envisioned for implementing the present invention include but are not limited to stand-alone computing environments and networked computing environments including intranets (i.e., local area networks (LANs), extranets including wide area networks (WANs) and the Internet.

Referring to Figure 3, a detailed description of a preferred embodiment of the present invention is shown. The EBMS 300 comprises a business process engine 304, a business management engine 308, a business rule engine 312, an event store 306, and system adapters 316 for communication with external systems 320.

In accord with the preferred embodiment of the EBMS, the business rule engine 312 is in operable communication with the business management engine 308, the business process engine 304, the event store 306, and system adapters 316. The business process engine is in operable communication with the event store 306, the business management engine 308, and the business rule engine 312. The business management engine 308 is in operable communication with the event store 306, the business process engine 304, and the business rule engine 312. In further accord with the preferred embodiment, the system adapters 316 operably connect the external systems 320, to the event store 306 and the business rule engine 312.

As shown in block 304, the business process engine comprises a plurality of process instances 324. Referring also to Figure 4, each process instance 324 or 400 comprises a plurality of work steps 404a-404*n* that are constituent to the overall process instance 324 or 400.

Preferably, at the beginning and end of each work step 404a-404*n*, a respective event 412a-412*n* is generated. Events 412a-412*n* comprise a time-stamped object of flexible structure and content that contains various business objects. Generally, events 412a-412*n* include information about changes in work step attributes 408a-408*n*. Exemplary business objects include but are not limited to a purchase order, a customer record or a document.

Referring again to Figure 3, the business process engine 304 i) presents process instance events 332 to the event store 306 for persistent storage, ii) presents process instance attributes 328 to the business management engine 308 for monitoring, and iii) presents events 332 to the business rule engine for processing. Additionally or alternatively, events 364 are presented to the event store 306 or business rule engine 312 by external systems 320 via the system adapters 316.

The event store 306 is configured to receive and store events generated by the business process engine 304 and the external systems 320, and output the stored events to the business rule engine 312 or the business management engine upon pushes or pulls. Preferably, the business rule engine 312 pulls stored events from the event store 332 according to a conventional producer-consumer model.

Preferably, the event store 306 is configured as a database table wherein each table record stores a single event object 332. Event object attributes stored within the database table include but are not limited to event type, value, date and context. The context attribute may be stored as a string of characters representing the attribute content (e.g., the set of all name / value pairs associated with this event) in a binary format.

In addition to the event object attributes, the database table maintains a unique event identifier attribute for tracking each event object 332. The event identifier attribute is not associated with the event object 332 itself, it is automatically attributed to any event 332 posted within the event store 306 by the event store. When the rule engine 312 pulls events 332 from the event store, it keeps track of the last event pulled by tracking its identifier.

The event store 306 also tracks the event identifier attribute assigned to the last event that was processed by the rule engine 312. Preferably, event store tracking is implemented utilizing an event counter (not shown). When the rule engine 312 has finished processing an event 332, it will update the tracking counter with the value of the event identifier maintained by the event store 306. If the rule engine 312 is shutdown and restarted, it will query the event store 306 to fetch the value of the event counter (indicating the last event processed before shutdown). Based on the fetched value, the rule engine 312 will process events having the next sequential event identifier.

Notably, the communication of events 332 from the business process engine 304 to the business rule engine 312 via the event store 306 allows the business process engine 304 to operate without effective knowledge of the existence of the business rule engine 312.

The business rule engine 312 includes business rules 336, a rule processor 340, and business metrics 352.

Business rules 336 comprise an encoded set of declarative statements for i) selecting events 332 from the event store 306, ii) monitoring business metrics 352, iii) modifying the processes flow in the business process engine 304, iv) generating events 314, and v) carrying out and enforcing business management policies. Referring also to Figure 5, business rules 336 and 500 can defined in a variety of manners including business logic rules 504 and business management rules 508. Generally, business logic rules 504 define the flow and operation of business processes with conventional logic. Business management rules 508, however, control *how* an underlying business process is to be managed. Additionally, business management rules 508 dynamically control how changes in a real-time underlying business process are to be carried out. As such, business management rules 508 operate superior to and generally separate from subordinate business logic rules 504.

An example of a business logic rule 504 relating to a purchase order process in as follows: "All purchase orders over $1000 should trigger a credit check." Essentially, this rule is part of the purchase order process definition. By contrast, a related business management rule 508 *decides* to raise this $1000 limit at a previously or automatically defined future date.

Business management rules 508 may implement business management policies in a variety of different manners including but not limited to the following:

- Defining business policies that control the conditions – or context – in which a business process executes (e.g., task assignment, request handling, resource allocation)
- Transitioning from one business policy to another (e.g., timing, scope of application, etc)
- Transitioning from the current version of an application to a new version,
- Enforcing some level of quality of service depending on the customer
- Generating and populating business reports

- Escalating business priorities
- Generating notifications of specific events

In accord with a preferred embodiment of the present invention, a business rule language programmatically encodes the business rules 336 into a computer-executable format. Generally, the business rules are executable functions comprising operational logic and business management policies having adjustable parameters. In further accord with the preferred embodiment, business rule parameters, as well as business metrics 352 (discussed in greater detail below), are defined based on a common data structure having adjustable attributes or parameters (e.g., the limit on a purchase amount before a credit check is required).

Notably, parameters associated with business rule functions 336 and business metrics 352 may be adjusted based on the execution of other business rule functions or user input presented at the business management engine 308 (discussed in greater detail below). For example, a business logic rule 504 in a help-desk process may decide to raise the priority of a user request from "high" to "critical" if the request is not completed within 2 hours. The time limit (2 hours) can be parameterized, so that it can be changed by simply changing the time limit parameter. As discussed previously, this change may also be executed based on user input at the business management engine 308 or another business rule function 336.

Additionally, parameter changes to business rule functions 336 and business metrics 352 may be executed based on the business metrics themselves. In accord with the previous example, the business metrics 352 might include statistics such as request completion time, resource availability or the level of customer satisfaction.

Business rules 336 may be defined and activated prior to the operation of the electronic business management system 300 or activated "on the fly" during the real-time operation of the invention. Notably, business rules 336 may be loaded or unloaded without interrupting the operation of the business process engine 304. Additionally, the manual or automatic parameterization of business rules 336 allows for their dynamic change without unloading and reloading the business rule 336 within the business rule engine 312.

Business metrics 352 are data concerning the dynamic aspects of an executing process instance 324 including but not limited to process instance events 332, the timing of process instances 324, the volume of concurrent process instances 324, data resulting from computation over attributes 328 of concurrent process instances 324, resource usage or availability, and communication with external systems 320.

As discussed in more detail below, the rule processor 340 dynamically updates business metrics based on events 332 generated by the business process engine 304 and external systems 320. Business metrics 352 are modified incrementally based on event 332 fetched from the event store 306 by the business rule engine 312. Additionally, the rule processor 340 may correlate several events 332 in order to update the business metrics 352.

As business metrics 352 are generated and updated, they are presented to the business management engine 308 or attached to business rules 336 and fed back to the rule processor 340. An example of a business metric 352 that is attached to a business rule 336 is as follows: "If the objective - to stay under 10% of late orders each month - is not reached, then the manager will be notified, and the current order handling policy may be disabled by another business rule." In this example, the business metrics 352 would be the monthly percentage of late orders.

The rule processor 340 applies business rules 336 and business metrics 352 to any combination of the following: i) events 332 pulled from the event store 306, ii) events 364 presented by external systems 320 via the system adapter 316, iii) events 314 generated by the rule processor 340 itself, iv) business metrics 352 for generating business reports 356, and v) events 332 fetched from the business process engine 304. In response to this application, the rule processor 340 brings about any combination of the following: i) control 318 over the business process engine 304, ii) action(s) 360 in external system(s) 320 via the system adapters 316,

iii) the generation of business metrics 352, or iv) the generation of events 314. Notably, business metrics 352 are generated and updated by the rule processor 340 dynamically during the real-time operation of the EBMS 300 for real-time presentation to the business management engine 308.

Generally, the rule processor 340 can fetch and monitor events 332 from several concurrent process instances 324. Applicable business rules 336 can i) correlate the several concurrent events, ii) consolidate data from these events, iii) calculate the time between the events, iv) modify relevant business metrics 352 or v) update relevant business rule parameters.

In accord with a preferred embodiment of the present invention, both business metrics 352 and rule parameter structures 353 are implemented in the same manner using data table structures (i.e., "infopads") of dimension 1, 2 or more. Table cells comprise an aggregation of atomic values of various types (e.g., character string, real number, integer number, etc.). Within each cell, the atomic values are identified by at least one attribute or "slot" name. In addition, each dimension of a table can be labeled. Consider the following two examples:

Example 1: A business report 356 on "products" comprises two table dimensions: each row of the table represents a product line and each column represents a sales region. Rows and columns are labeled, respectively, by product names and region names. Each cell of the products report maintains the following values: { quantity, total revenue, average_qty_per_order, maximum_qty }. All cells in this table have the same structure. Notably, these business reports could be generated each month and aggregated along the time dimension, each entry of which will be associated with a month of the year (such a report would be implemented utilizing a 3-dimensional infopad).

Example 2: A rule parameter structure 353 is configured to store, for each priority value (low, medium, high, critical) and each type of help-desk request event (e.g., software, hardware, equipment, installation, etc.), the time after which the event priority should be escalated. This time will be stored in each table cell. For example, if the cell <medium, equipment> has the value "2", then if the help-desk request has not been resolved after 2 hours it will be escalated to the next priority level (high). Here, each cell has a single attribute: {escalation_time}. Such a table of parameters represents a priority policy for help-desk request events and is implemented as an two-dimension infopad.

The infopad described in example #2 is used to parameterize a "priority_escalation" rule that will automatically escalate event priority if a predefined time limit is exceeded. In further accord with example #2, a two-rule process proceeds as follows:

- Rule #1 is triggered by the help-desk request event. The event contains a type of request (reqtype) attribute and a priority level (reqprio) attribute. Rule #1 queries the infopad for the corresponding time limit before escalating priority. In addition, rule #1 schedules a timeout event to execute at the time limit for the priority escalation.

- Rule #2 is triggered by the timeout event scheduled by rule #1. Rule #2 queries the infopad and determines whether the priority escalation has been completed. If not, the rule #2 escalates the priority of the help desk request to the next level. Additionally, rule #2 schedules another timeout event to execute at the time limit for the new priority.

Notably, the business report 356 and rule parameter structure 353 described in examples #1 and #2 are generated/implemented in the same manner. In addition, the business report 356 and rule parameter structure 353 are utilized in the same manner with respect to other rules. Accordingly, business rules can be parameterized by business metrics 352. Additionally, business rules 336 can change the parameters 353 of other business rules in the same manner they update business metrics 352.

In accord with a preferred embodiment of the present invention, control 318 is provided by a programming interface to the business process engine 304, which can be remotely called by the rule engine 312 using conventional remote procedure calls. Control functions provided in accord with the present invention include but are not limited to the timing of process instances 324, the definition of process instance attributes (Figure 4, 408a-408*n*), resource allocation, changing the value of an attribute of a business process instance, creating a new process instance, suspending or aborting a process instance, changing the priority of a process instance, forcing the completion of a process workstep, changing the assignee of a process workstep, changing the due date of a process instance, and resuming the execution of a process instance, either by a specific call to the business process engine or by changing a process attribute (causing a workstep of the business process to rest until the attribute takes an expected value).

Another aspect of the business rule engine 312 includes an event cache (not shown) that supports the correlation of multiple events that occur at different times. The event cache comprises a subset of events that (i) have been read or processed by the rule engine 312, and (ii) are relevant in some manner to one or more business rules (indeed, many events 332 and 364 may have no relevance to business rules 336). Once a relevant event has been read by the rule engine 312, it *is stored within the event cache.*

The event cache may indexed in several ways including: (i) by type and event attribute (i.e., there is an index entry for each type/value pair associating all events that possess the index value/pair); and (ii) by process instance identity attribute (indeed, each process instance attribute 408 has a unique identifier that is reported in every event 412 it generates). Both indexing schemes are supported jointly to organize the same event objects. Either one index or the other may be used by the rule engine 312 when executing a rule, depending on the rule profile.

Notably, a cache list for the event cache is maintained within the event store 306 persistent storage. If the EBMS 300 is restarted after shutdown, the event cache within the business rule engine 312 is reconstructed by copying all of the events within the event store 306 that have an identifier within the cache list.

Figure 8 is a block flow diagram illustrating a preferred event correlation sequence within the rule engine 312 where all of the correlated events belong to the same process instance, as the rule language allows for specifying this property in a rule. As represented in block 802, incoming events are matched with an event variable (e.g., Ek) of the correlation rule to be applied and executed. Next, the event cache entry is selected that contains all of the events generated by the current process instance, as represented by block 804. Notably, the process instance identity is reported in Ek. For each of the remaining event variables for the correlation rule being applied (i.e., E1 through E*n*, excluding Ek), find in the current index entry the most recent event in the event cache that: (i) matches the type and value of this event variable; and (ii) is different than the events that have already been selected for this rule, as represented in block 806. Once a combination of events from the event cache has been found that matches the event variables E1 through E*n* (excluding Ek), as represented in block 808, then the rule conditions are assessed for satisfaction as represented in block 810. If the rule conditions are satisfied, the correlation rule is executed as represented in block 812.

Figure 9 is a block flow diagram illustrating a preferred event correlation sequence within the rule engine 312 where all of the events do not come exclusively from the same process instance. They either come from a plurality of process instances or originate from external systems. As represented in block 902, incoming events are matched with an event variable (e.g., Ek) of the correlation rule to be applied and executed. For each remaining event variable of the correlation rule (i.e., E1 through E*n*, excluding Ek), the subset of events that relate to the same type and value are identified within the event cache (using the type/value indexing scheme), as represented in block 904. For each of the remaining event variables of the correlation rule (i.e., E1 through E*n*, excluding Ek), the most recent event that: (i) matches the type and value of the event variable; and (ii) is different from the events that have already been selected for the rule, are identified within each event's associated subset entry, as represented in block 906. Once a combination of events from the event cache has been found that matches the event variables E1 through E*n* (excluding Ek), as represented in block 908, then the rule conditions are assessed for satisfaction as represented in block 910. If the rule conditions are satisfied, the correlation rule is executed as represented in block 912.

In further accord with the preferred embodiment of the rule correlation aspect of the present invention, the rule engine 312 is configured to discard events from the event cache when the events are no longer needed. This function generally regulates the number of events contained within the event cache (as well as the cache list maintained in persistent storage within the event store). Although the size of the event cache may be substantial (depending on the quantity of extant events), the event cache will maintain a stable size and not grow continuously under normal working conditions of the present invention. Preferably, the number of event combinations that are tested for correlation are bound in a rule-based fashion.

Referring again to Figure 3, external actions 360 include but are not limited to accessing an external database for updating or retrieving information, generating an external alarm, starting up an external application, starting up adapters for existing applications, printing, or sending electronic mail messages. Actions 360 are performed by business rules 336 and involve the scheduling of an event to be generated at a future date. The event scheduler 365 contains a list of event records ordered by date of execution. Each time a new event is scheduled, it is inserted at an appropriate chronological position within the list of event records. A timer (not shown) operates within the event scheduler 365. When the execution date of the earliest scheduled event is reached, the timer triggers the event scheduler to forward the next event to the event store manager. Next, the timer is restarted for triggering the next event. The timer can also be configured to periodically check the date of the earliest scheduled event. In the event that the EBMS is shut down and restarted, all overdue events are automatically sent to the event store 306.

External systems 320 include but are not limited to databases, third-party software tools, applications, servers, or other hardware devices. Preferably, external systems 320 have an application programming interface, or "API" through which the system adapters 316 can operably communicate.

Preferably, communication between the business rule engine 312 and the distributed components of the EBMS 300 (e.g., control 318, actions 360, and business metrics 352) is event-based similar to the communication between the business process engine 304 and the business rule engine 312.

Preferably, a user of the EBMS 300 can define or select from a variety of pre-defined external actions 360 and system adapters 316. An exemplary system adapter 316 includes an MIB adapter that enables the business rule engine 312 to accept input from any SNMP agent, including agents that are part of Computer Associates' Unicenter TNG Enterprise Management System. Another adapter converts events generated by the Unicenter system to the EBMS format and vice-versa. Yet another system adapter 316 includes a mail server adapter that enables the EBMS components to send, receive, and monitor electronic mail messages.

As shown in block 308, the business management engine comprises business reports 356 and business managers 348.

Based on i) business metrics 352 generated by the business rule engine 312, ii) process instance attributes 328 presented by the business process engine 304, and iii) event store queries 310, business reports 356 are dynamically generated, populated, and updated for presentation to business managers 348. Via the business reports 356, business managers 348 can monitor business information including, but not limited to: i) the timing and attributes 328 of process instances 324 executing within the business process engine 304 and ii) statistical data resulting from the timing and attributes of a plurality of process instances 324. Essentially, the business reports 356 comprehensively emulate the present and past state of the overall business process being carried out by the EBMS 300 including transactions between distributed EBMS components, users, external servers and clients.

Preferably, business managers 348 can dynamically define business reports 356 via business rules 336 and view the business reports 356 at any time during the real-time operation of the EBMS 300.

Exemplary business reports 356 include but are not limited to the following:

- Employee Assignment Profile and Project Accounting - This report enables an accounting manager to determine the amount of time spent by an employee on specific projects.
- Time Off Statistics - This report enables a human resources manager to determine the amount of time off already taken by an employee year-to-date.
- Purchase Profile - This report enables a business manager to determine the total number of rejected purchase requests, as well as those over a certain amount, in the past six months.

Unlike conventional business reports, the EBMS business reports 356 and the business metrics 352 they are generated from are dynamically viewed, defined, and updated in real time while the business process engine 304 is running. Accordingly, the business reports 356 can be viewed, updated, and re-defined even before they are completed, which could take hours or days.

Preferably, business reports 356 are monitored by business managers 348 including human experts within a particular business domain. Exemplary business domains include accounting and human resources. Additionally, business metrics 352 can be monitored automatically by the business rule engine 312 as the business metrics 352 are being built in real time. Even before a report is completed, external actions 360 such as alarms and notifications can be sent if some abnormal values are detected within the business metrics 352, or if the progression of the business report 356 in time shows that some quantified objectives are at risk. Essentially, automatic metrics monitoring by the business rule engine 312 allows for a very early detection of arising problems – i.e. before the reports that would show this problem are even completed. Accordingly, system response time to a detected problem is minimized.

In response to the business reports 356, business managers 348 can dynamically i) view, create, update, discard, or define a timed implementation or suspension of business rules 348 within the business rule engine 312, and ii) view, create, update, discard, or define a timed implementation or suspension of process instances 324, process instance attributes 328, or work steps (Figure 4, 404a-404*n*) within the business process engine 304. Additionally, the business rule engine 312 may be configured to automatically implement or change business rules 336 in response to business metrics 352 directly (without intervention by business administration 348). An example of a business policy that is implemented based on a business metric 352 is: "If more than 10% of orders of preferred customers are late over this month, then re-route any new preferred order to the back-up server and skip the credit-checking step." Here, the monthly percentage of orders by categories of customers is obtained automatically and dynamically from a business metric 352.

Notably, business management activity including the monitoring and defining of business rules and process instance attributes is effected dynamically during the real-time operation of the EBMS 300 without causing the interruption of any business process instance 324 executing within the business rule engine 304.

Referring to Figure 6, an overview of a business rule definition is shown. Business rule definition 600 includes but is not limited to a rule name 604, a list of event profiles 606, an event condition 608, and an action list 612. Rule name 604 defines the name of the business rule 600. Each event profile 606 comprises an event type and event value referring respectively to the "type" and "value" attributes of an event. The vent profile 606 will allow a rule to select only events relevant to the rule before further testing the condition of relevant events. The event profile 606 lists all of the types of events expected to trigger the rule, based on their type and value attributes. When more than one event profile applies, the event profiles define the events having potential to be correlated. Combinations of events satisfying these profiles will then be tested according to event condition

608. Referring also to Figure 3, event condition 608 defines a condition to be satisfied by business events 332 and 364 fetched by the business rule engine 312 before the rule processor 340 can execute the action list 612. The event condition 608 might include i) logical conditions formed using operations such as AND, OR, and NOT and ii) management conditions or business policies. Action list 612 consists of one or more actions to be performed or policies to be implemented in a specified sequence if the event condition 608 is satisfied. Actions 612 include but are not limited to generating reports, populating business metrics, escalating business priorities, generating notifications of specific events, loading or unloading other rules, changing parameters of rules, automatically processing or assigning tasks, dynamically monitoring and managing workloads and resources, and controlling process instances executing within the business process engine 324.

An exemplary business rule definition includes the following:

Rule Name: Preferred_Customer_Late_Shipment

Event Condition: A high priority order from a preferred customer has not been shipped within 24 hours of manufacture.

Action List: Automatically raise process priority to critical; notify on-duty supervisor; and log information into the management report.

Referring to Figure 7, business event definition is shown. Business event definition 700 includes but is not limited to event type 704, event date 708, event value 712, and event context 716.

Event type 704 is used as a main discriminator for the business event 700. Referring also to Figure 3, the event type 704 identifies either the source of the of the event (i.e. business process engine 304 or system adapter 316), or the type of a major event such as a "New Subscription."

Event date 708 includes a timestamp for the event 700. Preferably, the precision level of event date 708 ranges from days to milliseconds.

Event value 712 is a parameter regarding the event 700 that contains business data. Event value 712 is a second-level discriminator for the business event 700. For example, the value 712 of a business event 700 generated by the business process engine 304 depends upon the type of action 704 that generated the event 700. Preferably, event value 712 is pre-defined for a given event type 704.

Referring also to Figure 3, event values 712 include but are not limited to the following: the creation, installation, and removal of a business process within the business process engine 304; the creation, activation, suspension, completion, removal, prioritization, and dead-lining of a process instance 324a-324*n*; the activation, termination, completion, prioritization, and dead-lining of a work step (Figure 4, 404a-404*n*); the updating of data; the state of an external system 320; and the dead-lining, beginning, and ending of general business events 332 and 364.

Event context 716 contains a list of name-value pairs specific to an event type 704 and event value 712. Event context 716 represents any additional business data of interest for the event 700. For example, event context 716 may contain a mapping of complex business objects that business managers (Figure 3, 348) wish to associate with the event 700, such as a purchase order.

Preferably, event type 704, event date 708, event value 712, and event context 716 are reported to business managers (Figure 3, 348) in the form of an event map textually defining each event definition.

While embodiments of the invention have been illustrated and described, it is not intended that these embodiments illustrate and describe all possible forms of the invention. Rather, the words used in the specification and appendix are words of description rather than limitation, and it is understood that various changes may be made without departing from the spirit and scope of the invention.

WHAT IS CLAIMED IS:

1. A computer-based system for dynamically managing electronic business processes, the system comprising:

a business process engine configured to:

i) host the execution of a plurality of electronic business processes; and

ii) generate an event object upon the execution of each electronic business process wherein the event object reflects the current state of the business process; and

a business rule engine configured to host the execution of a plurality of business rule functions wherein the business rule functions are triggered based upon the event objects and are configured to:

i) generate business metrics based on the event objects; and

ii) dynamically exercise control over the electronic business processes based on the event objects and the business metrics.

2. The system of claim 1 wherein the business rule functions are configured to exercise control over the timing and attributes of electronic business processes executed within the business process engine.

3. The system of claim 1 additionally comprising a business management engine configured to populate at least one electronic report based on the business metrics wherein the electronic report is populated in real-time without interrupting the operation of the business process engine.

4. The system of claim 1 additionally comprising a persistent storage device configured to store event objects generated by the business process engine for subsequent transmission to the business rule engine.

5. The system of claim 1 wherein the business rule engine is additionally configured to receive event objects generated by an external system.

6. The system of claim 1 wherein the business rule engine is additionally configured to receive user input modifying a business rule function wherein the modification takes place in real-time without interrupting the execution of business processes within the business rule engine.

7. The system of claim 1 wherein the business metrics and parameters of the business rule functions are each defined using a common data structure that can be operated upon interchangeably by business rule functions.

8. The system of claim 7 wherein at least one business rule parameter is adjusted by the execution of at least one business rule function.

9. The system of claim 7 wherein at least one parameter is adjusted based on user input.

10. The system of claim 1 wherein at least one of the business rule functions is configured to generate an event object.

11. The system of claim 1 wherein at least one of the business metrics is configured to generate an event object.

12. The system of claim 1 wherein at least one of the business rule functions are configured to exercise control over an external system.

13. The system of claim 1 additionally comprising an event scheduler configured to transmit an event object to the rule engine at a predefined date and time.

14. The system of claim 1 wherein the business process engine is additionally configured to suspend the execution of at least one electronic business

process until the execution of at least one business rule function has been completed.

15. The system of claim 1 wherein the business rule engine is configured to correlate more than one event object over time and execute a business rule function based on the correlation.

16. The system of claim 1 wherein the system is implemented over a computer network.

17. A computer-implemented method for dynamically managing electronic business processes, the method comprising:

executing a plurality of electronic business processes; and

generating an event object upon the execution of each electronic business process wherein the event object reflects the current state of the business process;

executing a plurality of business rule functions wherein the business rule functions are triggered based upon the event objects;

generating business metrics based on the event objects; and

dynamically generating control over the electronic business processes based on the event objects and the business metrics.

18. The method of claim 17 wherein the business rule functions are configured to exercise control over the timing and attributes of electronic business processes executed within the business process engine.

19. The method of claim 17 additionally comprising populating least one electronic report based on the business metrics wherein the electronic report is populated in real-time without interrupting the execution of the electronic business processes.

20. The method of claim 17 additionally comprising storing the event objects in persistent storage for subsequent retrieval.

21. The method of claim 17 additionally comprising receiving an event object generated by an external system.

22. The method of claim 17 additionally comprising receiving user input modifying a business rule function wherein the modification takes place in real-time without interrupting the execution of the electronic business processes.

23. The method of claim 17 wherein the business metrics and parameters of the business rule functions are each defined using a common data structure that can be operated upon interchangeably by the business rule functions.

24. The method of claim 23 additionally comprising adjusting business rule parameters based on the execution of at least one business rule function.

25. The method of claim 23 wherein at least one parameter is adjusted based on user input.

26. The method of claim 17 wherein at least one of the business rule functions generates an event object.

27. The method of claim 17 wherein at least one of the business metrics generates an event object.

28. The method of claim 17 additionally comprising dynamically generating control over an external system based on the event objects and the business metrics.

29. The method of claim 17 wherein a business rule function is triggered based upon an event object presented at a predefined date and time.

30. The method of claim 17 additionally comprising suspending the execution of at least one electronic business process until the execution of at least one business rule function has been completed.

31. The method of claim 17 additionally comprising correlating more than one event object over time wherein a business rule function is executed based on the correlation.

32. The method of claim 17 wherein the method is implemented over a computer network.

33. A system for dynamically managing electronic business processes, the system comprising:

a means for executing a plurality of electronic business processes; and

a means for generating an event object upon the execution of each electronic business process wherein the event object reflects the current state of the business process;

a means for executing a plurality of business rule functions wherein the business rule functions are triggered based upon the event objects;

a means for generating business metrics based on the event objects; and

a means for dynamically generating control over the electronic business processes based on the event objects and the business metrics.

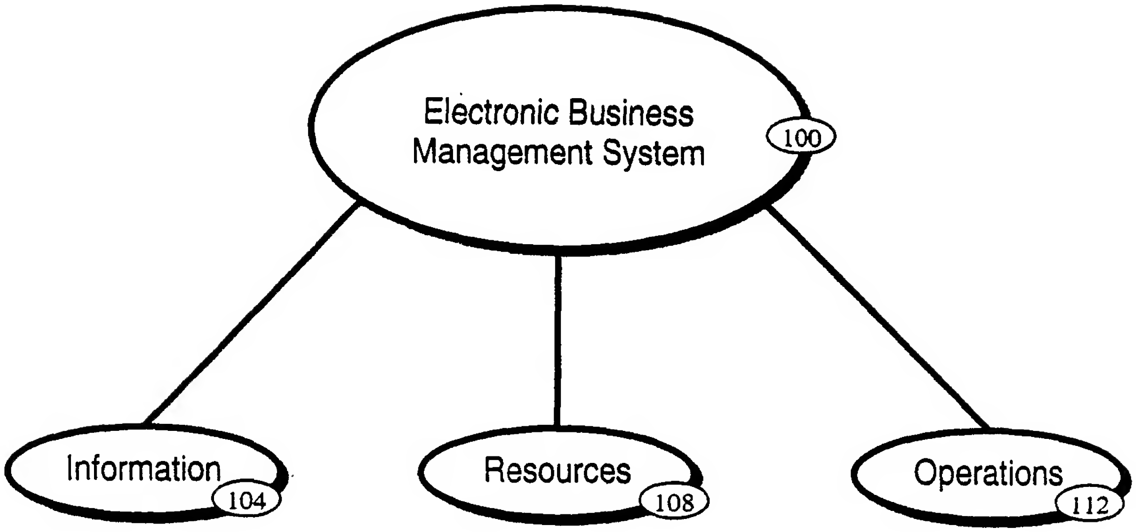

Electronic Business Management System
100
Information
104
Resources
108
Operations
112

Fig. 1

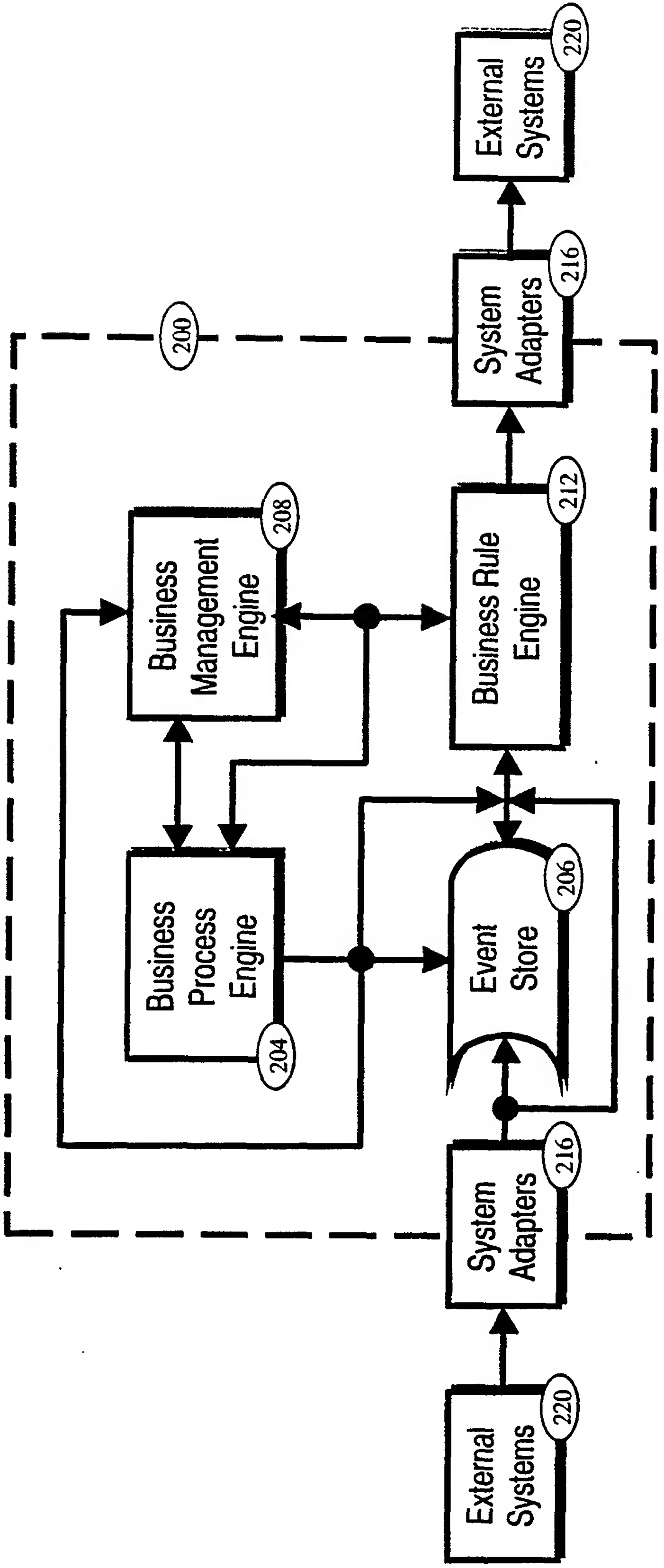

External Systems
220
System Adapters
216
200
Business Management Engine
208
Business Rule Engine
212
Business Process Engine
204
Event Store
206
System Adapters
216
External Systems
220

Fig. 2

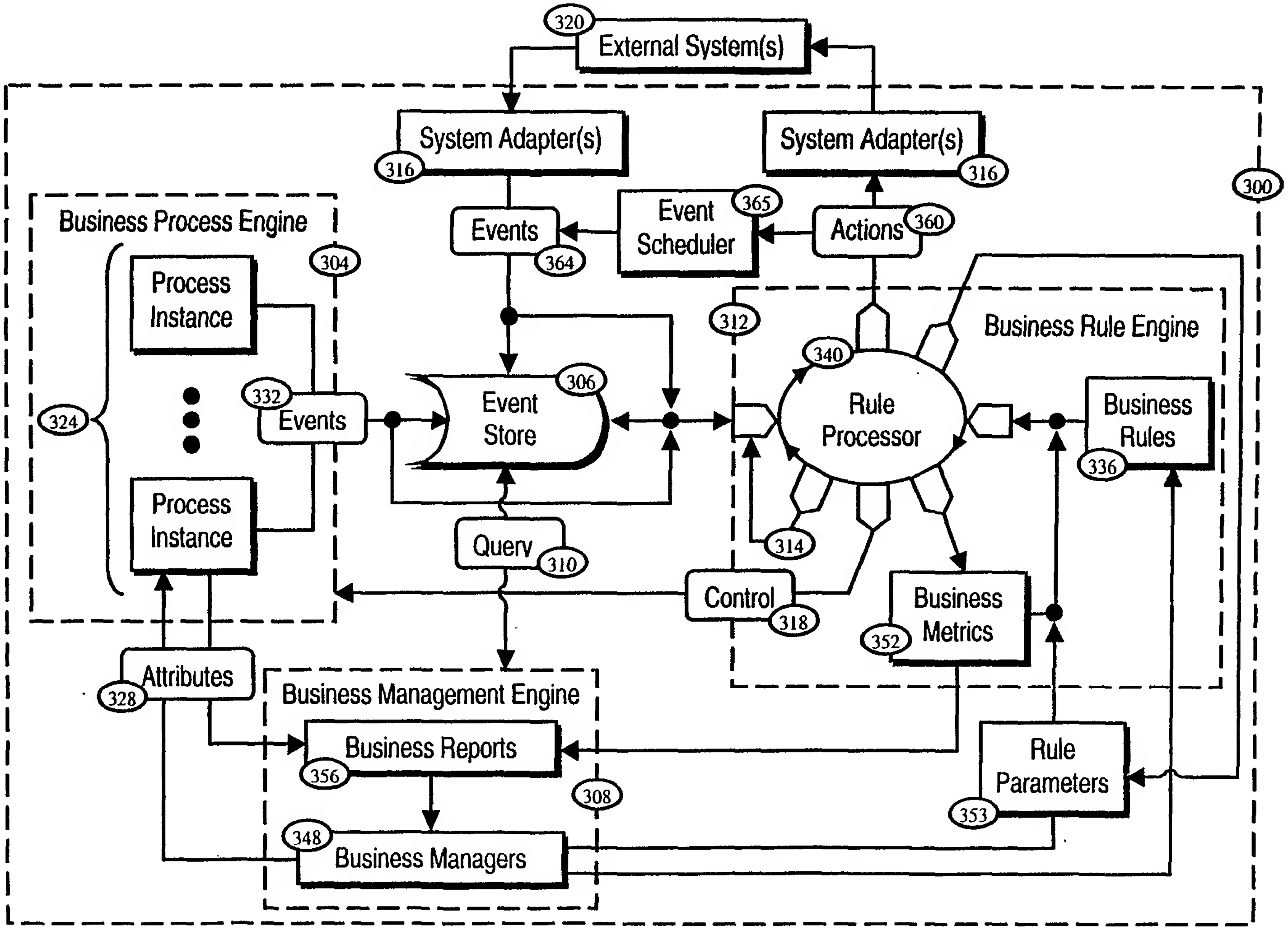

320
External System(s)
System Adapter(s)
316
System Adapter(s)
316
300
Business Process Engine
304
Process Instance
Process Instance
324
332
Events
Events
364
Event Scheduler
365
Actions
360
312
Business Rule Engine
340
Rule Processor
Business Rules
336
Event Store
306
Querv
310
314
Control
318
Business Metrics
352
Attributes
328
Business Management Engine
Business Reports
356
308
Business Managers
348
Rule Parameters
353

Fig. 3

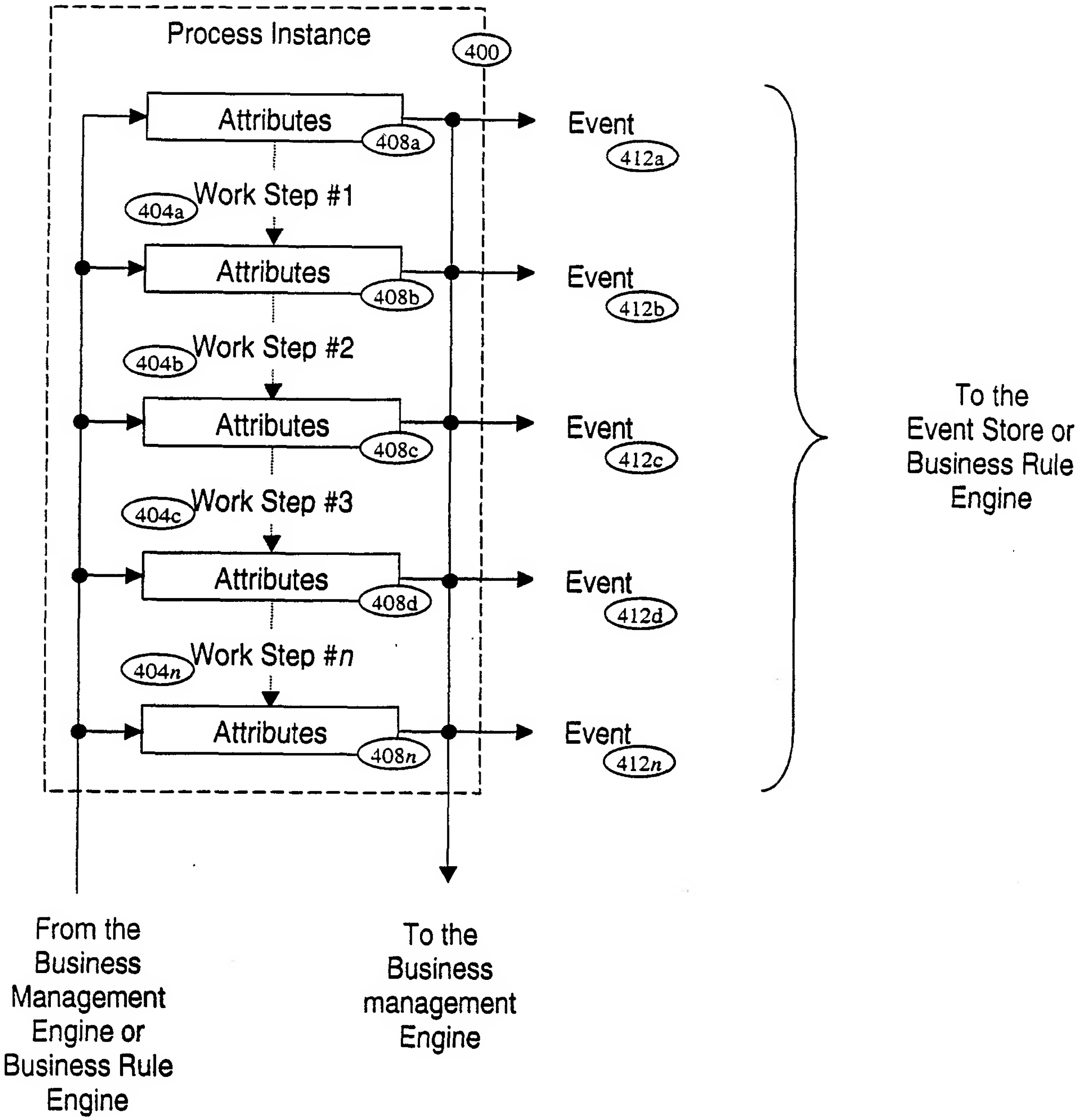

Process Instance
400
Attributes
408a
Event
412a
404a
Work Step #1
Attributes
408b
Event
412b
404b
Work Step #2
Attributes
408c
Event
412c
404c
Work Step #3
Attributes
408d
Event
412d
404n
Work Step #n
Attributes
408n
Event
412n
To the Event Store or Business Rule Engine
From the Business Management Engine or Business Rule Engine
To the Business management Engine

Fig. 4

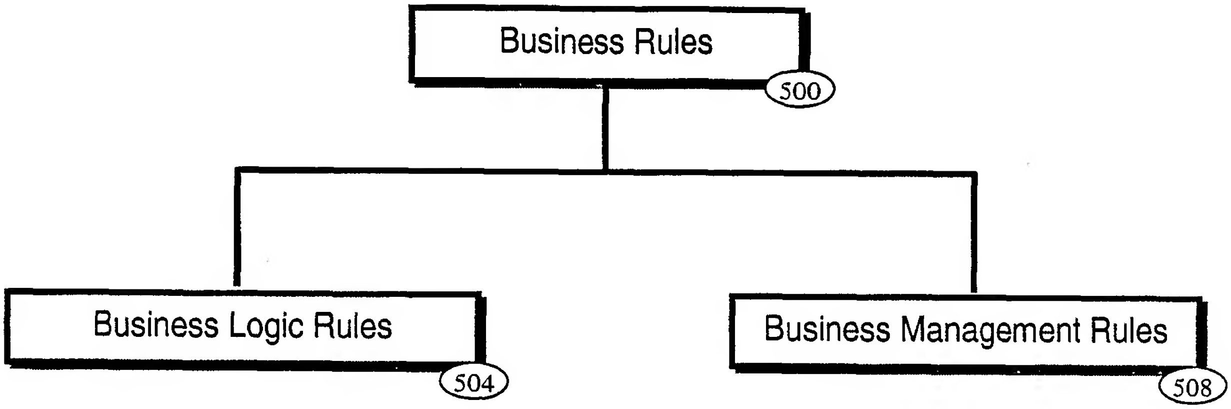

Business Rules
500
Business Logic Rules
504
Business Management Rules
508

Fig. 5

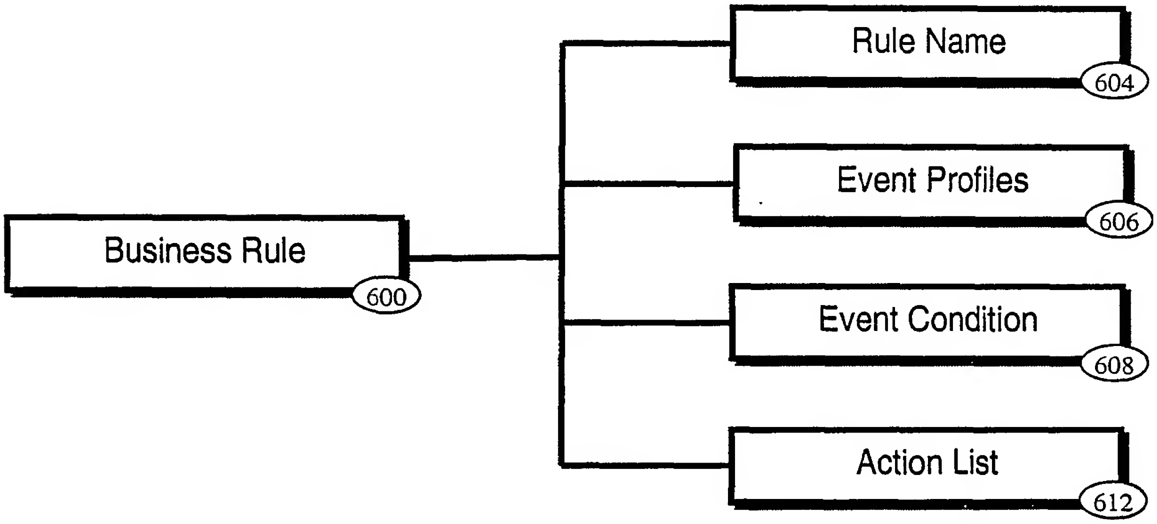

Business Rule
600
Rule Name
604
Event Profiles
606
Event Condition
608
Action List
612

Fig. 6

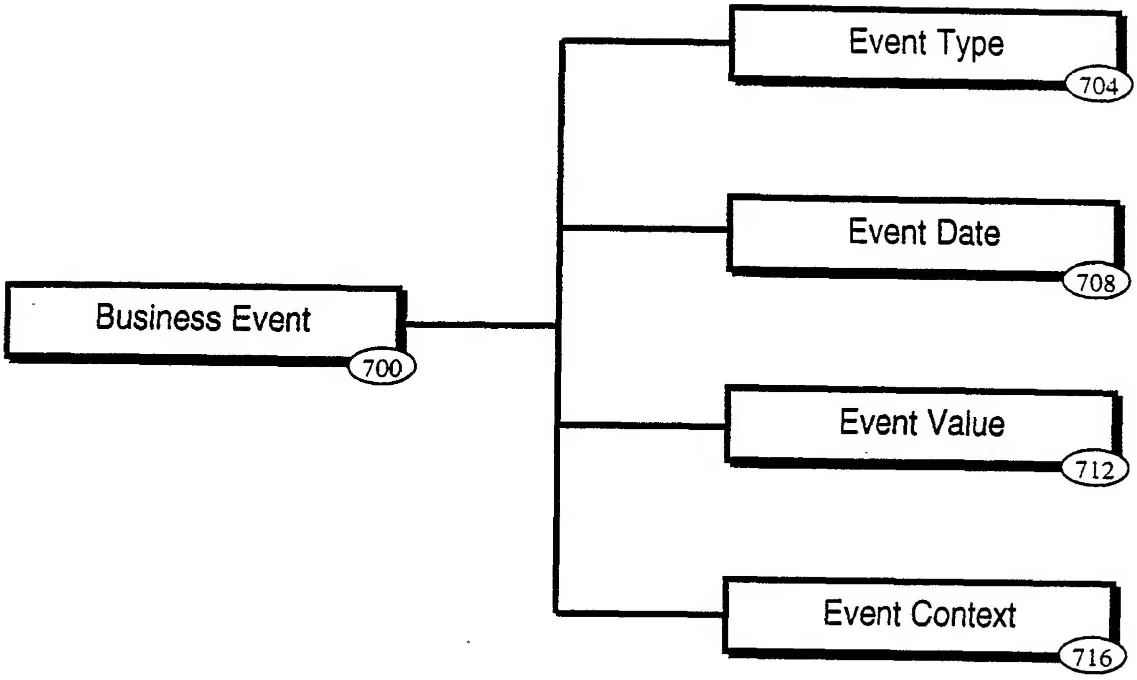

Business Event
700
Event Type
704
Event Date
708
Event Value
712
Event Context
716

Fig. 7

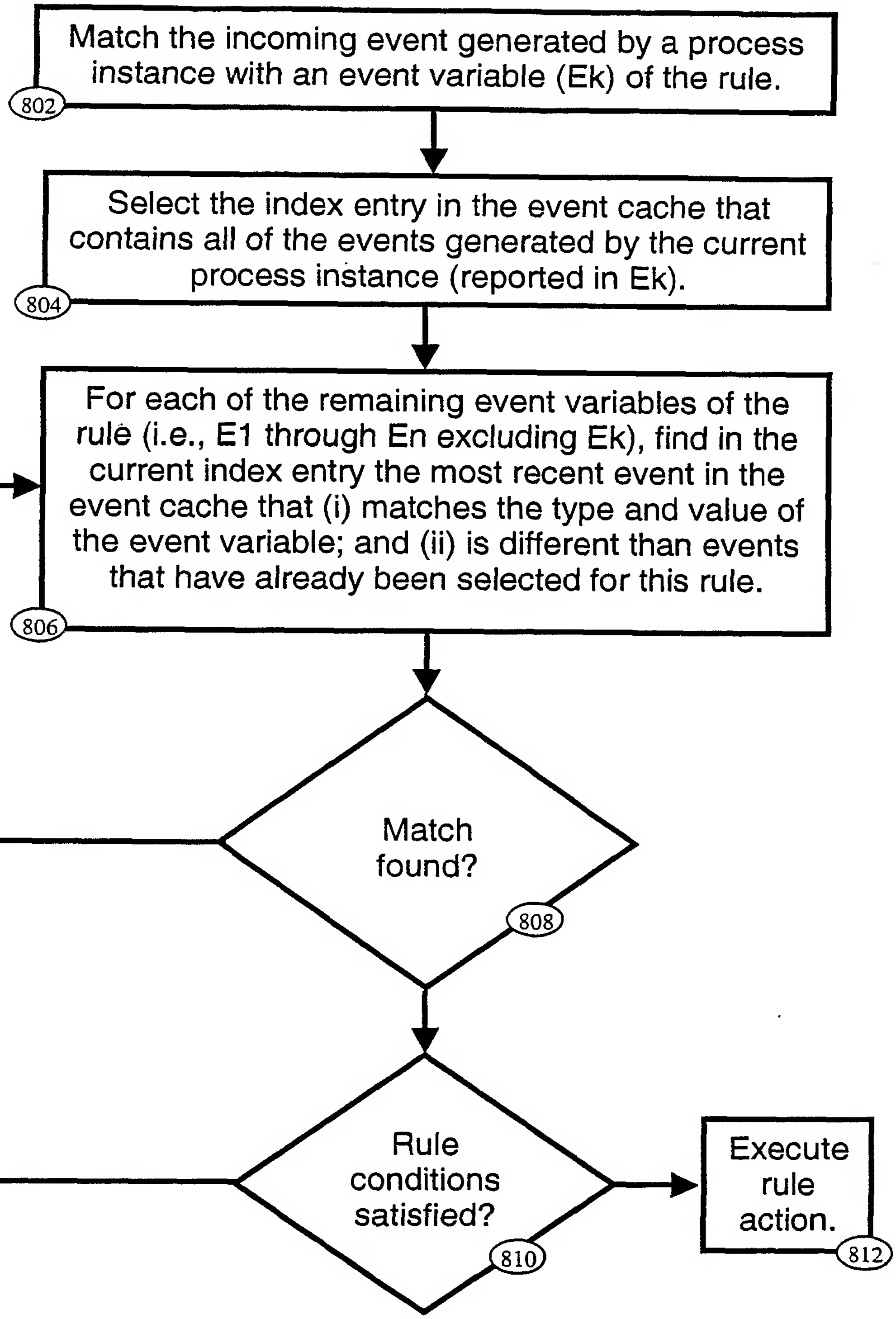

Match the incoming event generated by a process instance with an event variable (Ek) of the rule.
802
Select the index entry in the event cache that contains all of the events generated by the current process instance (reported in Ek).
804
For each of the remaining event variables of the rule (i.e., E1 through En excluding Ek), find in the current index entry the most recent event in the event cache that (i) matches the type and value of the event variable; and (ii) is different than events that have already been selected for this rule.
806
Match found?
808
Rule conditions satisfied?
810
Execute rule action.
812

Fig. 8

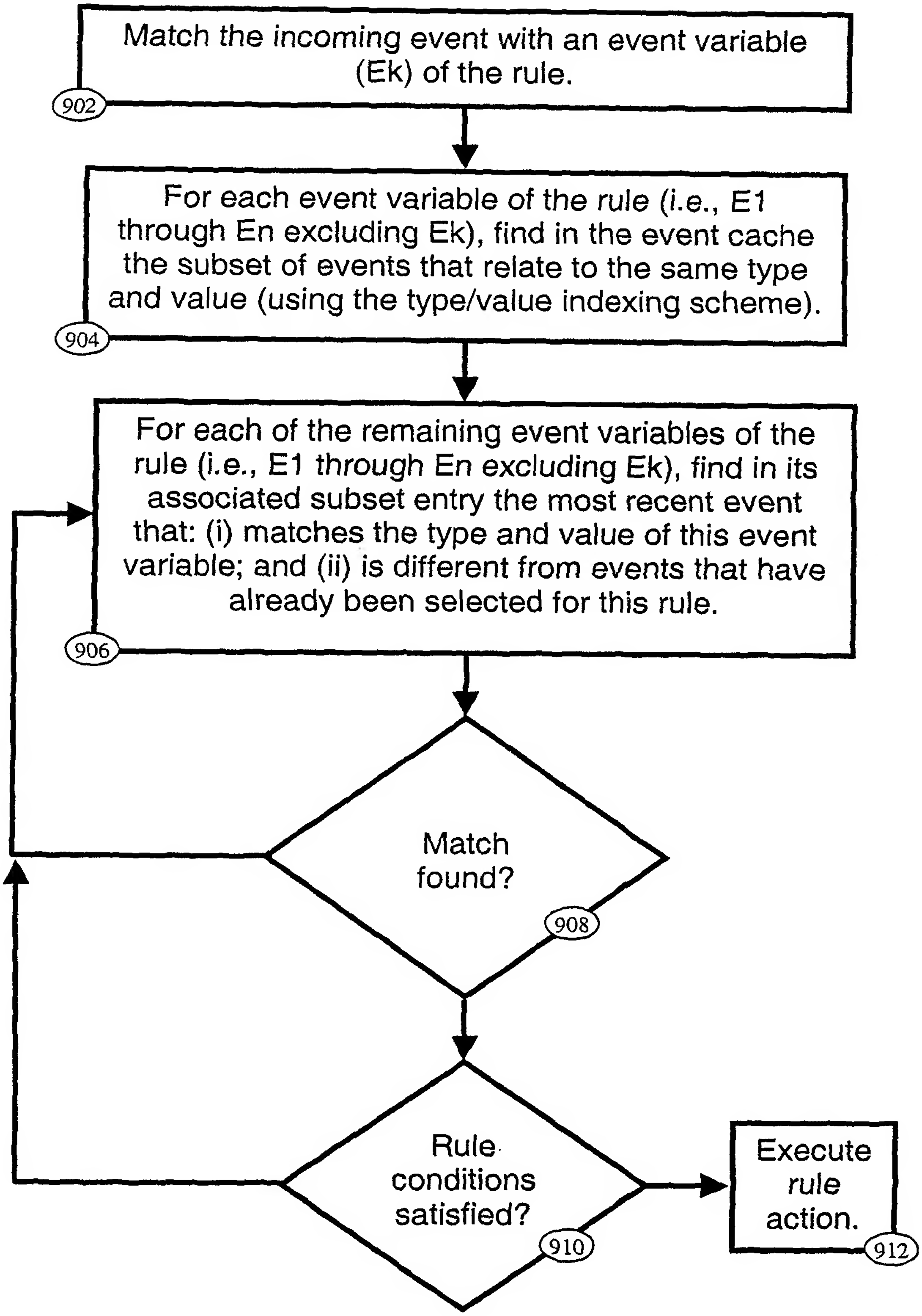

Match the incoming event with an event variable (Ek) of the rule.
902
For each event variable of the rule (i.e., E1 through En excluding Ek), find in the event cache the subset of events that relate to the same type and value (using the type/value indexing scheme).
904
For each of the remaining event variables of the rule (i.e., E1 through En excluding Ek), find in its associated subset entry the most recent event that: (i) matches the type and value of this event variable; and (ii) is different from events that have already been selected for this rule.
906
Match found?
908
Rule conditions satisfied?
910
Execute rule action.
912

Fig. 9